プログラミング学習支援ツール pgtracer を自学習に活用したプログラミング科目の実践報告

村田美友紀[†1]　嘉藤直子[†2]　掛下哲郎[†3]

我々は，プログラミングの初学者を対象としたプログラミング学習支援ツール pgtracer の開発を行っている．pgtracer は Moodle 上で動作し，プログラムとトレース表に対する穴埋め問題を出題する．また，pgtracer は学生の学習ログを自動収集し，学生の学習行動や理解度を分析する機能も提供する．我々は，2016 年度と 2017 年度にプログラミング科目において，pgtracer を活用した自学習課題を提供した．2016 年度の実践では，学生の利用が少なく，トレース表の理解が十分ではなった．そこで，2017 年度は pgtracer の自学習課題を小テストや定期試験に含め，その成果を成績に反映させた結果、学生の利用状況が改善した．また，pgtracer の説明時間を増やし，授業中の説明や演習にトレース表を用いた結果，トレース表の理解が向上した．本稿では，その実践報告を行うとともに，pgtracer が収集したログを用いて学生の学習行動について検討する．

キーワード：コンピュータプログラミング教育，e-learning，Moodle，穴埋め問題，ラーニングアナリティクス

A Practical Report on Self-Learning utilizing pgtracer at an Actual Programming Class

MIYUKI MURATA[†1]　NAOKO KATO[†2]
TETSURO KAKESHITA[†3]

Abstract: We are developing a programming education support tool pgtracer for programming beginners. Pgtracer is running on Moodle and provides fill-in-the-blank questions. A question is composed of a C++ program and a trace table. Moreover, pgtracer automatically collects learning log of the students and provides data analysis functions for the collected log. The data analysis functions will support teachers to recognize learning activity and achievement of each student and the entire class. We assigned homework utilizing pgtracer to the students taking a programming course at National Institute of Technology, Kumamoto college in 2016 and 2017. In 2016, the number of students utilizing pgtracer was not many and we found many students whose understanding of the trace table is not enough. We included the fill-in-the-blank questions in mini tests and examinations in 2017 and reflected the learning achievement of the student to the student's evaluation of the programming course. As a result, student's learning activities are greatly improved. Furthermore, we spend more time for the explanation to introduce pgtracer and use trace tables in the lecture and exercise. As a result, student's understanding of the trace table was improved. In this paper, we report self-learning utilizing pgtracer at an actual programming class. Then we consider the student's activity utilizing the student's log collected by pgtracer. .

Keywords: computer programming education, e-learning, Moodle, fill-in-the-blank question, learning analytics

1. はじめに

プログラミングは，高専や大学など工学系の学生にとって重要である．2020 年度より小中学校および高校でのプログラミング教育が順次開始される予定になっており，プログラミング教育の重要性はますます高まっている[1].

ところが，高専や大学における実際のプログラミングの授業では,学力や学習意欲が低い学生がしばしば見られる．プログラミング上達のためには，多くのプログラムを作成することが有用だが,授業時間数や教師数などの制約から，授業時間だけでは十分な演習を行うことができない．これを補うためには学生の自学習による取り組みが必要である．しかし，自主性に任せるだけでは，学生は学習しない傾向にあり，また学生の自習状況を教員が把握できない．課題とする場合にも，採点や提出状況の確認など，教員にかかる負担は大きい.

我々は, Moodle のプラグインとして動作するプログラミング学習支援ツール pgtracer を研究開発している[2]．pgtracer はプログラムとトレース表に対する穴埋め問題を提供する．インターネットが利用できる PC があれば，学生は好きなときに pgtracer を利用してプログラミング学習ができる．また，pgtracer の自動採点機能は教員の採点の負荷を軽減し，データ分析機能は教員による学生の学習状況の把握を支援する.

pgtracer を自学習に適用することによって，学習時間の確保，変数のトレースやプログラムに対する理解の向上，変数や関数の名前付けやインデンテーションなどが適切なプログラムを読む機会の増加が期待できる.

†1 熊本高等専門学校
National Institute of Technology, Kumamoto College
†2 有明工業高等専門学校
National Institute of Technology, Ariake College
†3 佐賀大学
Saga University

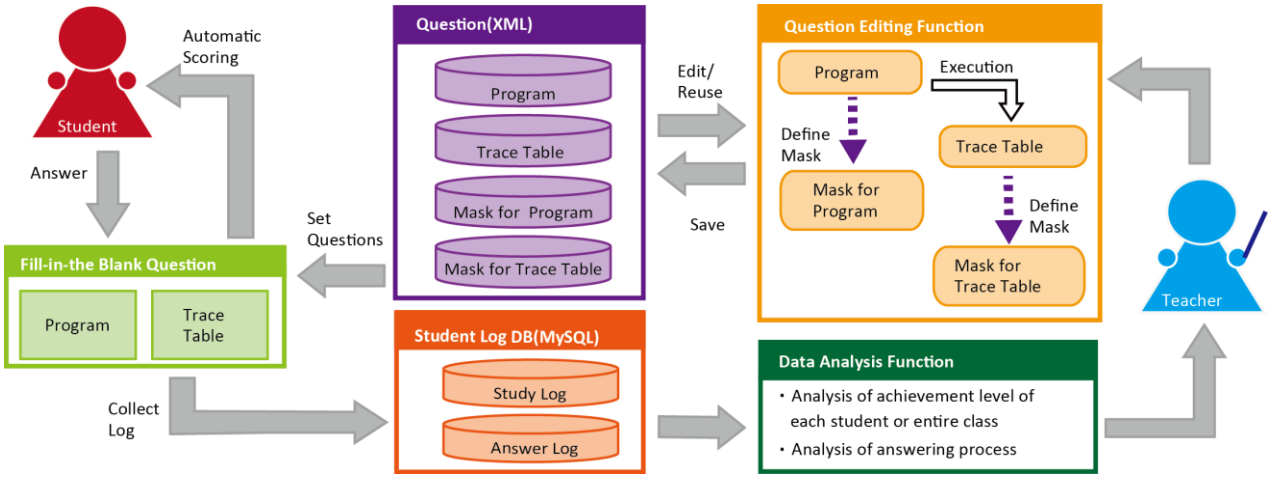


図 1　pgtracer を用いたプログラミング教育プロセス

Figure 1　Programming Education Process utilizing pgtracer.

本稿では, 2016 年度と 2017 年度に熊本高等専門学校 (以下, 熊本高専) 八代キャンパスの授業で実践した自学習課題の提供についての実践報告を行う[3, 4]. 2016 年度の実践では pgtracer を活用する学生が少なかった．また，トレース表の概念を理解できず，トレース表の穴埋めに困難を感じる学生が多く見られた．これを踏まえていくつかの改善を行った結果，2017 年度は，学生の取り組み状況やトレース表の理解が向上した．また，自学習課題の解答数が多い学生ほど定期試験の成績が良いことが分かった．

本稿は以下のように構成されている．2 節では，pgtracer の概要について述べる．3 節では，学生が履修するプログラミング科目について説明する．4 節と 5 節では，2016 年度と 2017 年度の実践における自学習課題の提供方法や問題作成方針について述べる．6 節では，実践の結果について考察する．最後にまとめと今後の課題について述べる．

2. pgtracer の概要

2.1 教育プロセス

図 1 に pgtracer を用いたプログラミング教育プロセスを示す．まず，教員はあらかじめ作成した問題プログラムと入力ファイルをもとに，pgtracer を用いて穴埋め問題を作成する．穴埋め問題は，プログラムとトレース表から構成される．プログラムに設定できる穴抜きは，トークンまたは文であり，トレース表に設定できる穴抜きは，変数値，ステップ，変数名である．これらの穴抜き個所の組み合わせによって，問題難易度を調節できる．

学生は，問題一覧から問題を選び解答する．pgtracer には自習モードと試験モードがある．自習モードでは学生が穴埋めをすると即時にその正誤が判定され，穴抜きの塗りつぶしの色によって学生が正誤を確認できる．試験モードでは学生が解答を提出すると pgtracer が正誤の判定を行い，結果を表示する．

pgtracer はログ分析機能を提供しており，学生は学習履歴や利用者全体の平均や分布を閲覧など，自身の学習状況を確認できる．教員は，学生ごと，問題ごとに正解率や解答所要時間，穴抜きごとの学生の解答を確認できる．また学生の解答手順を再現することができ，学生個人の指導や授業へのフィードバックに利用できる．

2.2 システムの機能

pgtracer が出題する穴埋め問題は，プログラムとトレース表からなる（図 2）．トレース表はプログラム実行のそれぞれのステップにおける変数と出力の値を示している．穴埋め問題は，プログラム，トレース表，プログラム用マスク，トレース表用マスクの 4 つの XML ファイルから構成される．プログラムとプログラム用マスクを分離することによって，1 つのプログラムに対し，難易度の異なる複数の問題を作成できる．トレース表も同様の理由でマスクを分離している．

ステップ	プログラム
	//キーボードから10個の整数を入力し，その合計と平均を表示する．
	//平均は小数点第2位まで表示する
	# include < stdio.h >
	int main ( ) {
1	int i ; //繰り返しの制御用変数
2	int num , sum = ＿＿ ; //入力した数，合計を格納
3	＿＿＿＿ avg ; //平均を格納
	//整数の入力と合計への加算
4	＿＿＿＿ (i = 0 ; i < 10 ; i ++) {
4.1	scanf ("%d" , &num) ;
4.2	sum ＿＿＿＿ num ;
	}
	//平均の計算
5	avg = (＿＿＿＿) sum / 10 ;
	//結果の表示
6	printf ("合計＿%d\n" , sum) ;
7	printf ("平均＿% 2f\n" , avg) ;
8	return 0 ;
	}

ルーチン	ステップ	main i	main num	main sum	main avg	出力
main	1	?				
main	2	?	?	0		
main	3	?	?	0	?	
main	4	＿＿	?	0	?	
main	4.1	＿＿	11	0	?	
main	4.2	＿＿	11	11	?	
main	4	＿＿	11	11	?	
main	4.1	＿＿	22	11	?	
main	4.2	＿＿	22	33	?	
main	4	＿＿	22	33	?	
main	4.1	＿＿	33	33	?	
main	4.2	＿＿	33	66	?	
...中略...						
main	4	9	99	495	?	
main	4.1	9	110	495	?	
main	4.2	9	110	605	?	
main	5	＿＿	110	605	60.5	
main	6	＿＿	110	605	60.5	合計 605
main	7	＿＿	110	605	60.5	平均 60.50
main	8	＿＿	110	605	60.5	

図 2　pgtracer が出題する問題

Figure 2　A Fill-in-the-Blank Question Provided by pgtracer.

表 1　熊本高専におけるプログラミング教育科目
Table 1　The Programming Class in National Institute of Technology, Kumamoto College.

科目名	開講時期	単位
2015 年度以前入学		
基礎情報工学	2 年　通年	2
マイコンプログラミング	2 年　後期	1
プログラミング基礎	3 年　通年	2
2016 年度以降入学		
プログラミング基礎 I	1 年　後期	1
プログラミング基礎 II	2 年　前期	1
マイコンプログラミング	2 年　後期	1
情報処理	3 年　通年	2

学生が穴抜きを埋めるたびに，pgtracer は解答，正答，解答所要時間をログとして収集する．pgtracer は収集したログの分析機能として，学習履歴一覧，学生ごとの分析機能，問題ごとの分析機能，穴抜きごとの分析機能，解答プロセス分析機能を提供する．

pgtracer で設定できる問題オプションは，出題モード（自習用/試験用），正答の表示/非表示，問題分析機能の表示/非表示，穴抜き選択時の当該ステップの色付けの有無がある．出題モードの自習用では，学生が解答を入力するたびに，正解プログラムとの文字列比較により，その正誤を判定する．ここで，正答が正解プログラムで記述された文字列 1 つだけではない場合，即時に正誤の判断はできない．そこで，pgtracer は，解答の提出後に穴埋めしたプログラムをコンパイル・実行し，トレース表と比較する．これによって，pgtracer は正解プログラムと一致しない解答であっても正しく正誤を判定できる．

3. 連携するプログラミング科目

3.1　熊本高専八代キャンパスにおけるプログラミング教育カリキュラム

熊本高専八代キャンパスには，機械知能システム工学科，建築社会デザイン工学科，生物化学システム工学科があり，被験者となる学生は情報工学を専門としないが，すべての学生に対して共通教育科目としてプログラミング教育を行っている（表 1）．また，専門科目としても各専門に応じたプログラミング科目が設置されている．

熊本高専では 2016 年度にカリキュラムが改定され，年度進行でカリキュラムが移行されている．旧カリキュラムの「基礎情報工学」は，新カリキュラムの「プログラミング基礎 I」と「プログラミング基礎 II」に対応する．また，新旧対応科目で使用する教科書は同じである．

2016 年度には，旧カリキュラムの「基礎情報工学」の後期において，2017 年度は，新カリキュラムの「プログラミング基礎 I」において，pgtracer による自習課題を提供した．

表 2　基礎情報工学の授業スケジュール
Table 2　Lecture Plan of Fundamental of Computer Science

前期		後期	
週	内容	週	内容
1	コンピュータの基本	1	1 次元配列
2	数値の表現	2	1 次元配
3	フローチャート	3	2 次元配列
4	定数，変数，代入	4	2 次元配列
5	printf, scanf	5	ポインタ
6	型と演算	6	ポインタ
7	条件分岐	7	関数
8	前期中間試験	8	後期中間試験
9	条件分岐	9	関数
10	for	10	関数
11	while	11	変数のスコープ
12	do while	12	ファイル
13	break, continue, switch	13	構造体
14	課題実習	14	構造体
15	前期末試験	15	学年末試験

表 3　2016 年度に出題された課題
Table 3　Assignment of the Class in 2016

No	問題
1	身長（整数）と体重（実数）を入力し，標準体重を表示
2	2 つの整数を入力し，大きい数から小さい数を引いた値を表示
3	整数を入力し，「正」，「負」，「0」のいずれかを表示
4	3 つの整数値を入力し，最大値を表示
5	整数を入力し，その値によって表示を変える．（フローチャートは提示）
6	点数（整数）を入力し，評価（A, B, C, D，入力エラー）を表示
7	段数を入力し，＊を使ってその段数のピラミッドを表示
8	入力した文字列の中の大文字，小文字，数字の数を表示
9	整数を入力し，桁数の逆順から表示
10	整数を入力し，その桁数を表示

3.2　基礎情報工学（2016 年度実施）

本科目は 2 名の教員が担当する．主担当が授業および課題を含めた成績評価を担当し，副担当は演習のサポートを行う．著者の一人が生物化学システム工学科の副担当を担った．基礎情報工学の達成目標は，以下の 3 項目である．

(1) **コンピュータの基礎知識を説明できる**
(2) **Visual C++の開発環境を使うことができる**
(3) **基本的なプログラムを作成できる，**

授業スケジュールを表 2 に示す．pgtracer による自学習課題の提供は後期に実施した．

評価方法は試験 80%，課題 20%である．課題は通年で 10 個出題され，プログラム，フローチャート，実行結果の提出とともに口頭試問が行われた．出題された課題を表 3 に示す．諮問に合格すれば，それぞれ 100 点として評価する．授業は，教員による説明の後，学生が各自でサンプルプログラムの動作確認や課題を各自で取り組む方式であった．

試験は紙ベースで実施され，後期中間試験に pgtracer で出題した問題の 1 問を部分的に出題した．その配点は後期中間試験全体の 4%であった．

3.3 プログラミング基礎 I（2017 年度実施）

本科目は 2 名の教員が担当し，主担当を著者の一人が担った．プログラミング基礎 I の達成目標は，以下の 5 項目である．プログラムを理解するためには，変数の動きがトレースできることが重要と考え，達成目標に追加した．

(1) **コンピュータ内での数値や文字の表現方法を説明でき，問題に適したデータ構造を設計できる**

(2) **C プログラムの作成から実行までの処理ができる**

(3) **変数と代入，標準入出力を用いたプログラムが作成できる**

(4) **条件分岐を含むプログラムを作成でき，変数の動きをトレースできる**

(5) **反復処理を含むプログラムを作成でき，変数の動きをトレースできる**

本科目を履修する学生は，情報工学を専門としていない．このため，データ型は int 型と double 型，char 型のみを扱うなど教育内容を絞った．1 回の授業では，3〜4 課題の演習を実施し，ペアワークにより演習内容を相互確認した．

また，トレース表の理解を促進するため，授業において変数の変化を説明する際にトレース表と同様の表を用いた．

評価方法は試験 60%，課題 20%，小テスト 20%である．試験は，本キャンパスで利用可能な e-learning システムを用いた．2 回の試験のそれぞれで，pgtracer を用いた自学習課題の中から 2 問ずつを選び，問題のキャプチャ画像を用いて出題した．これらの問題の試験全体における割合は 10%である．小テストは，自学習課題の中から 3 問を選んで，授業中に pgtracer を利用して実施した．

同じ問題を出題することで，自学習課題への取り組みが良い学生は，小テストや試験において高い得点が取れるような仕組みとし，学生の自学習課題へのモチベーションの向上を図るとともに pgtracer の自学習課題への取り組みを科目の成績に反映する．

課題は 2 個出題し，プログラム，フローチャート，実行結果の提出とともに，口頭試問を行った．口頭試問ではプログラムの理解に加え，変数の名前やインデントなどのプログラミングスタイルについても確認を行い，諮問に合格すれば 100 点として評価する．出題した課題の内容は以下の通りである．

(1) **2 つの抵抗値と接続方法（直列または並列）を入力して，その合成抵抗を求める．接続方法は s または p の文字で指定する．**

(2) **5 つの実数を入力し，printf('*');を用いて入力した実数に対応する数の*を表示する．**

授業スケジュールを表 4 に示す．また，課題や小テスト，アンケート，自学習課題の問題数も併せて示す．

表 4　プログラミング基礎 I の授業スケジュール
Table 4　Lecture Plan of Fundamental of Programming I

後期			
週	内容	問題数	備考
1	プログラムの実行	-	pgtracer のユーザ登録
2	定数，変数，代入	2	pgtracer の説明
3	printf，演算子	2	第 1 回アンケート
4	フローチャート	3	
5	scanf, getchar	3	第 1 回小テスト
6	条件分岐（for）	3	課題 1
7	前期中間試験	-	
8	条件分岐	3	第 2 回アンケート
9	switch	2	
10	条件分岐の入れ子	3	
11	for 文	3	第 2 回小テスト
12	while 文，do while 文	3	課題 2
13	break, continue	3	
14	課題実習	2	
15	前期末試験	-	
16	まとめ		第 3 回アンケート

4. 2016 年度基礎情報工学における pgtracer を活用した自学習課題の提供

4.1 狙いと問題作成の方針

pgtracer を活用した自学習課題の提供には，以下の狙いがある．学習時間の確保，変数のトレースやプログラムに対する理解の向上，変数や関数の名前付けやインデンテーションなどが適切に記述されているプログラムを読む機会の増加．これらの狙いを達成するために，2016 年度の実践では，問題作成方針を以下のように定めた．

(1) **自習モードとする．**

(2) **1 回の課題の問題数は 3 つとする．**

(3) **問題は授業の内容に合わせ，いくつかのプログラムは授業で用いたプログラムと似たものとする．**

(4) **プログラムの穴抜きはトークンまたは文の一部分とする．**

(5) **トレース表については，前後のトレース表の値がヒントとなるため，問題とする穴抜きの前後のステップも合わせて穴抜きとする．**

(6) **変数名，コメント，インデントを適切に記述し，手本となるプログラムを提示する．**

（1），（2）は，学生の継続的な利用によって学習時間を確保するために定めた．2016 年度以前の経験により，pgtracer の自習モードで正誤が即時に判明することが，学生の学習意欲を高めることが分かっている．

（3），（4）の方針により変数のトレースや基本的な文法の使い方を確認する問題とする．pgtracer は，プログラムとトレース表を併記しており，学習者が穴抜きを選択すると，対応するステップ行が色付け表示されるため，処理の流れの理解を支援できる．

表 5　2016 年度に pgtracer により提供した自学習課題
Table 5　Questions Assigned using pgtracer in 2016 Class

問題	内容
(1)-1*	入力した 2 つの整数値の和，差，積，除，剰余を表示
(1)-2*	入力した整数値の絶対値と偶数か奇数かを表示
(1)-3*	入力した整数の 1 乗から 5 乗までを表示
(2)-1	入力した 10 個の整数値の合計と平均を表示
(2)-2	10 個の整数を入力した順に配列に格納し，逆順に表示
(2)-3	大文字なら小文字，小文字なら大文字に変換して表示
(3)-1	正数が 5 個になるまでに入力された負数の個数を表示
(3)-2	2 つの文字を連結して連結後の文字列を表示
(3)-3	2 次元配列に格納された座標のうち原点から最も遠い座標を表示
(4)-1	2 の 0 乗から 8 乗を配列に格納し，指定された整数乗の答を表示
(4)-2	”end”と入力されるまで入力した文字列の長さを表示
(4)-3	2 次元配列に格納された三角形の辺の長さの合計を表示
(5)-1	(3)-3 と同じプログラムで変数のコメントなし
(5)-2	2 次元配列に格納された 3 つの座標のうち原点から最も遠い点の座標を表示
(5)-3*	ポインタを使って変数に代入，表示
(6)-1	(4)-3 と同じプログラムで変数のコメントなし
(6)-2	ポインタを使って，円の半径から面積を求め，表示
(6)-3	ポインタを使って，配列要素を指定し，要素を表示
(7)-1	整数 n を入力して，n 回表示．n を引数とする関数を使う．
(7)-2	3 つの座標を入力して原点からの距離を表示．距離を求める関数を使う
(7)-3	関数を使って累乗を計算する
(8)-1	(3)-1 と同じプログラムでコードのコメントなし
(8)-2	名前と得点を入力し，得点に応じてメッセージを表示．メッセージ表示は関数を使う
(8)-3	いくつかの関数の中から適切な関数の呼び出し
(9)-1	階乗を求める関数を使って，順列を求め表示
(9)-2	秒から（時，分，秒）へ変換，（時，分，秒）から秒へ変換する．関数を使って求める．
(9)-3	配列のソート．配列要素の入れ替えは swap 関数を使う
(10)-1	(6)-2 と同じプログラムで変数のコメントなし
(10)-2	(9)-1 と同じプログラムでコードのコメントなし
(10)-3	配列を使った宝探しゲーム．あたりを判定する関数を使う．

＊は，テストユーザーを含む

表 6　2017 年度に pgtracer により提供した自学習課題
Table 6　Questions Assigned using pgtracer in 2017 Class

問題	内容
(1)-1	関連のない代入文
(1)-2	給与支給額を計算する
(2)-1	関連のない代入文
(2)-2	成績の表示
(3)-1	3 桁の各位の数の合計
(3)-2	面積と円周の長さの計算
(4)-1	○時○分を分，時間に換算
(4)-2	平均を求める
(4)-3	関連のない入力文
(5)-1	文字列の指定した場所の文字を変換
(5)-2	代金を求める
(5)-3	関連のない if 文
(6)-1	成績を判定する
(6)-2	得点から階級を求める
(6)-3	会員資格，回数から入場料を求める
(7)-1	月から四季を求める（else-if)
(7)-2	四則演算子
(8)-1	月から四季を求める（switch)
(8)-2	関連のない for 文
(8)-3	整数の和
(9)-1	入力した，正と負の数を数える
(9)-2	*を使った図の表示
(9)-3	11 段から 19 段の九九を表示
(10)-1	整数 n について，2^n を求める
(10)-2	while と do-while の違い
(10)-3	0 以下の実数が入力されるまでに入力された実数の和を求める
(11)-1	正の数を 5 個入力するまでに入力した負の数の個数を求める
(11)-2	入力した数の絶対値を求める．0 を入力したら終了
(11)-3	10 個の実数を入力し，正の数の個数と合計を求める
(12)-1	*を使った図の表示
(12)-2	整数の各桁を足し，1 桁になるまで繰り返す

プログラムを作成するうえで，適切な名前付けやインデンテーションは重要だが，授業内の説明だけでは学生の定着が難しい．そこで,手本となるプログラムを多く読ませることで，適切な名前付けやインデンテーションの習得を目指す．このために，(6) の方針を定めた．

自学習課題を提供している中で，変数名や関数名が長い場合，トレース表が大きくなり，問題全体を閲覧できないことが分かった．そこで，(6) の方針を考慮したうえで，変数や関数の名前を短くする，配列の要素数を少なくするなどの工夫を行なった．

4.2　問題の作成

自学習課題の問題作成および問題難易度やコメントの確認は，著者らのグループが行った．問題プログラムは授業の内容や進度に合わせたものを採用した．

10 回分の授業について, 全 30 問の問題を作成した (表 5)．1 回あたりの自学習課題 (3 問) を作成するのに要した時間は平均 2 時間であった．内訳は，問題プログラムの作成，プログラムやトレース表のマスク作成に各 1 時間であった．

4.3　自学習課題の提供

2016 年度は，最初の 2 週間の授業にて最後の 15 分ずつを使って pgtracer へのユーザ登録および pgtracer の使い方を説明した．しかし，時間内にユーザ登録が終了しない学生やトレース表の概念が理解できない学生がおり，時間不足であった．その後は完全な自学習課題とし，授業中に自学習課題を行なう時間を設けなかった．また，解答期限も設定しなかった．自学習課題の公開は学内メールで連絡し，授業において, 自学習課題実施の呼びかけは行っていない．

表 7　問題（6）-1 の情報

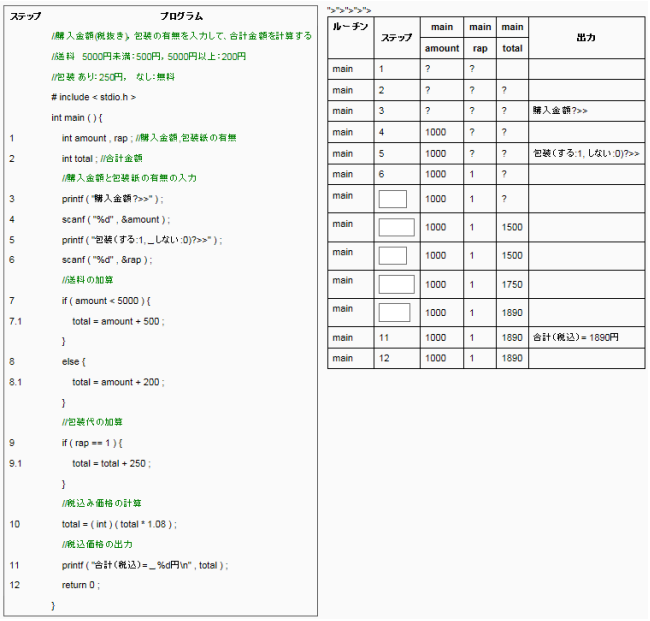
Table 7　Information of Question(6)-1

内容	商品の値段から送料を計算
狙い	if 文の動作がトレースできる
穴抜きの場所（個数）	トレース表（5）

表 8　問題（9）-2 の情報

Table 8　Information of Question(9)-2

内容	月を入力して，四季を表示
狙い	switch 文が書ける
穴抜きの場所（個数）	プログラム（5）



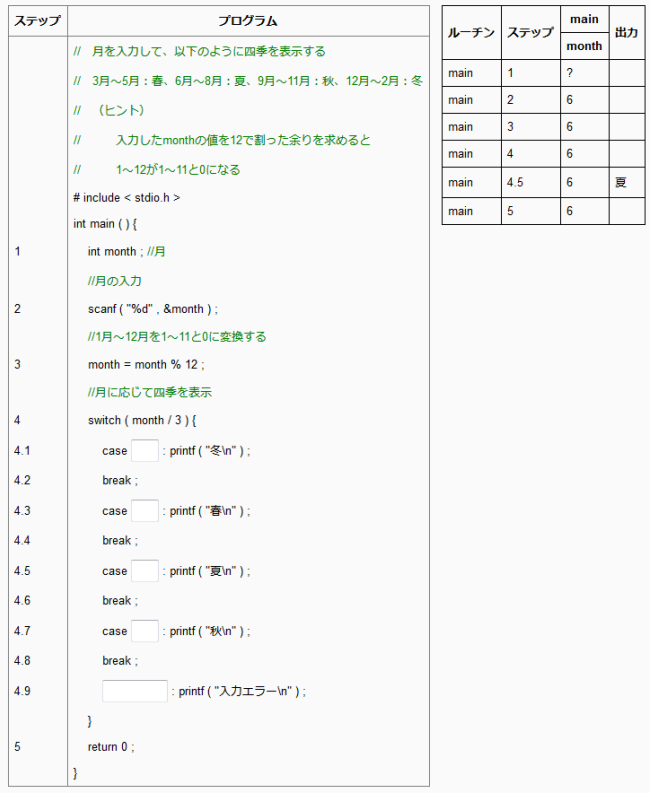
ステップ　プログラム

```
//購入金額(税抜き)，包装の有無を入力して，合計金額を計算する
//送料　5000円未満：500円，5000円以上：200円
//包装 あり：250円，　なし：無料
# include < stdio.h >
int main ( ) {
1     int amount , rap ; //購入金額 ,包装紙の有無
2     int total ; //合計金額
      //購入金額と包装紙の有無の入力
3     printf ( "購入金額?>>" ) ;
4     scanf ( "%d" , &amount ) ;
5     printf ( "包装(する:1, _しない:0)?>>" ) ;
6     scanf ( "%d" , &rap ) ;
      //送料の加算
7     if ( amount < 5000 ) {
7.1       total = amount + 500 ;
      }
8     else {
8.1       total = amount + 200 ;
      }
      //包装代の加算
9     if ( rap == 1 ) {
9.1       total = total + 250 ;
      }
      //税込み価格の計算
10    total = ( int ) ( total * 1.08 ) ;
      //税込価格の出力
11    printf ( "合計(税込)=_%d円\n" , total ) ;
12    return 0 ;
}
```

ルーチン	ステップ	main amount	main rap	main total	出力
main	1	?	?		
main	2	?	?	?	
main	3	?	?	?	購入金額?>>
main	4	1000	?	?	
main	5	1000	?	?	包装(する:1, しない:0)?>>
main	6	1000	1	?	
main		1000	1	?	
main		1000	1	1500	
main		1000	1	1500	
main		1000	1	1750	
main		1000	1	1890	
main	11	1000	1	1890	合計(税込)= 1890円
main	12	1000	1	1890	

図 3　問題（6）-1

Figure 3　Question (6)-1



ステップ　プログラム

```
//  月を入力して，以下のように四季を表示する
//  3月～5月：春，6月～8月：夏，9月～11月：秋，12月～2月：冬
//  (ヒント)
//      入力したmonthの値を12で割った余りを求めると
//      1～12が1～11と0になる
# include < stdio.h >
int main ( ) {
1     int month ; //月
      //月の入力
2     scanf ( "%d" , &month ) ;
      //1月～12月を1～11と0に変換する
3     month = month % 12 ;
      //月に応じて四季を表示
4     switch ( month / 3 ) {
4.1       case    : printf ( "冬\n" ) ;
4.2       break ;
4.3       case    : printf ( "春\n" ) ;
4.4       break ;
4.5       case    : printf ( "夏\n" ) ;
4.6       break ;
4.7       case    : printf ( "秋\n" ) ;
4.8       break ;
4.9           : printf ( "入力エラー\n" ) ;
      }
5     return 0 ;
}
```

ルーチン	ステップ	main month	出力
main	1	?	
main	2	6	
main	3	6	
main	4	6	
main	4.5	6	夏
main	5	6	

図 4　問題（9）-2

Figure 4　Question (9)-2

4.4　アンケート

最後に pgtracer による自学習課題に関するアンケートを実施した．アンケートの項目は，基礎情報工学の課題に要する 1 週間当たりの平均時間，その他の課題に要する 1 週間あたりの平均時間，解答した問題数，問題の難易度，pgtracer はプログラミング学習に有用か，課題に取り組まなかった理由である．

5.　2017 年度　プログラミング基礎 I における pgtracer を活用した自学習課題の提供

2016 年度の実践によって，プログラムとトレース表の穴抜きが混在する問題は，学生が難しいと感じることが分かった．このため，2017 年度の実践では，学生の負荷軽減を狙い，以下の方針も加えた．また，出題の意図を明確にするために，問題ごとに狙いを設定した．

(1)　**1 つの問題における穴の数を少なく設定する．**

(2)　**基本的事項を確認するための問題は，どちらか一方のみの穴抜きとする．**

トレース表の穴抜きについては，問題としたい変数値以外にも，その前後などに穴抜きを設定している．これは，ヒントとなることを避けるためである．これらの穴抜きについては，穴抜き数としてカウントしない．

2016 年度と同様に著者のグループによって，問題作成と妥当性の確認を行った．問題プログラムは授業内容に沿ったプログラムを扱った．12 回分の授業について，1 回あたり 2～3 問で全 32 問の問題を作成した．出題した問題の一覧を表 6 に示す．また，問題(6)-1，(9)-2 について作成した問題（図 3，図 4）とその情報（表 7，表 8）を示す．

2017 年度は，pgtracer へのユーザ登録および pgtracer の使い方の説明は，それぞれ 5 分増やして 20 分ずつとした．これにより，すべての学生が時間内にユーザ登録を完了できた．また，トレース表の概念を理解できない学生に対応するため，授業内で変数の変化の説明や演習問題にトレース表を用いることで，トレース表の概念の理解向上を狙った．また，授業時間内に pgtracer の問題に取り組む時間を確保できた回があった．

自学習課題の公開は授業中に連絡し，あわせて自学習課題の実施を呼びかけた．解答期限は次の授業までとした．解答期限内に実施しなかった場合のペナルティーはない．

学期の初期，中間試験後，期末試験後の計 3 回，アンケートを行った．アンケート項目は，pgtracer の使い方の理

表 9　問題解答数と試験成績の変化
Table 9　Average Difference of Exam Score

2016 年度				
解答数	人数	前期末	後中間	差
すべて	15	68.53	71.00	2.47
1～半数	36	74.67	75.56	0.89
0	51	71.43	71.25	-0.18
2017 年度				
解答数	人数	後中間	学年末	差
すべて	97	79.85	75.34	-4.51
1～半数	7	67.57	61.14	-6.43
0	1	92.00	82.00	-10.00

表 10　自学習課題の初回解答時と最高点時の平均点
Table 10　Average of Right Answer Ratio of the First Attempt and the Attempt with the Highest Score.

	2016 年度	2017 年度
初回解答時平均点	62.6	84.6
最高得点時平均点	82.2	99.0
解答回数の平均	1.50	1.85

解度，トレース表の理解度，問題の難易度，穴埋めの量，1 回の自学習課題に要した平均時間，その他の課題に要した 1 週間あたりの平均時間，プログラミングへの興味や意欲などである．

6. 考察

6.1 学生の学習行動に関する考察

図 5 は，各自学習課題の平均解答率を示している．ここで平均解答率とは，1 回の自習課題で出題された 2 問ないし 3 問の解答率の平均を示す．2016 年度は 10 回，2017 年度は 12 回自学習課題を出題している．2016 年度は解答率が低く，さらに回を追うごとに解答率が減少した．後期中間試験で pgtracer の問題を範囲としたため，後期中間試験前の自学習課題（6）までは解答する学生がいたが，それ以降は急激に減少し，すべての問題に回答した学生は 1 名であった．一方，2017 年度は，最も解答率の低いのが最後の自学習課題（12）で 80%と，徐々に減少する傾向はあるが，ほとんどで 90%を超えている．

解答履歴を精査すると解答所要時間が 2 分以下で得点が 0 点の解答が多く見つかった．これらの解答が試験直前に実施されていることから，試験に備えて実際には解答せずに，正答を確認していると考えられる．このような解答の割合は，2016 年度の 16.4%から 2017 年度は 5.6%と減少している．また，2017 年度の実践では，授業中においても演習を早く終わらせた学生が，自主的に pgtracer の自習課題に取り組む様子も観察できた．

以上より，2017 年度では学生の解答行動が改善されたことが分かる．これより，学生の学習意欲を高め継続させるためには，科目の成績に自学習課題の取り組みを反映させること，授業内で実施に対する声掛けを継続的に行うことが有効であることが分かった．

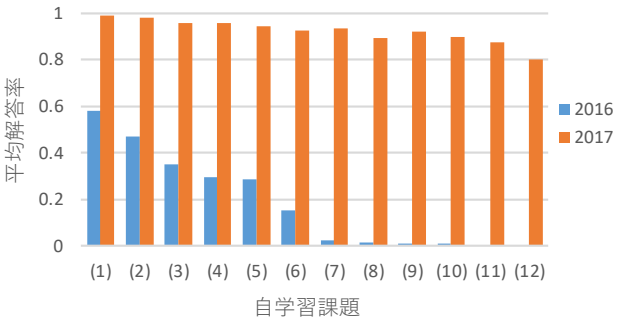


図 5　自学習課題ごとの平均解答率
Figure 5　Average of the answer ratio of each week.

表 9 に，問題解答数および 2 つの試験の成績変化の平均を示す．2016 年度は，後期中間試験と前期期末試験，2017 年度は，学年末試験と後期中間試験である．両年度とも多くの問題を解答した学生の得点が高いことが分かる．pgtracer の問題を解答することで，学習時間が増えたことが成績の向上につながったと考えることができる．

2016 年度では，半数以下を解答した学生の方が試験の平均点が高い．アンケートの回答より，「プログラムを理解しているので，pgtracer の自学習課題が必要なかった」と回答した学生が見られるなど，成績上位の学生が自学習課題に取り組まなかったことが原因と考えられる．

表 10 に自学習課題に出題した問題の初回解答時と最高得点時の平均点および解答回数の平均を示す．2016 年度は解答率が 10%を超えている自学習課題（1）から自学習課題（6）までを対象とする．初回解答時の平均点に比べ，最高得点時の平均点が 2016 年度で 19.6 点，2017 年度で 14.6 点ほど向上していることが分かる．また，解答回数の平均を見ると 2016 年度は 1.50 回，2017 年度は 1.85 回とほぼ同等の結果が得られた．なお，2016 年度，2017 年度ともに解答回数についての指示は行っていない．このことから，いったん自学習課題に取り組むことができれば，学生は高得点を目指して問題の解答を繰り返すことが分かった．つまり，その問題を理解するための行動をとることが分かった．

6.2 トレース表の理解に関する考察

2016 年度のアンケートでは，トレース表に関する項目は設けなかったが，自由記述欄において「トレース表の見方が分かりづらかった」，「トレース表に何を解答すればよいのかが分からなかった」などトレース表が何を意味しているのか，その概念の理解が不足していると推測できる記述があった．また，学生へのインタビューの中でも，「トレース表の見方が分からなかったのでトレース表の穴抜きは解答していない」，「最初はトレース表が分からなかった」と

の回答があった.

そこで, 2017 年度ではトレース表の理解を向上するための取り組みを行った. これを評価するために 2017 年度のアンケートでは, トレース表の理解に関する項目を追加した. 図 6 より，1 回目のアンケート時でも 80%の学生が理解できたと回答している.

以上より，開始時に pgtracer の使い方やトレース表の説明時間を増やしたことや，授業の中でトレース表を用いて説明や演習を行うことが，学生のトレース表の理解に役立ったことが分かった．また，授業においてペアワークによって学生同士の教えあいが行われ，教員が特に理解が不足している学生に対応できたことも有効であったと考える.

6.3 pgtracer の機能に関する考察

pgtracer は，学生ごと，問題ごとの解答履歴を一覧表示する機能を持つ．2017 年度には pgtracer の分析機能を用いて解答状況をモニターし, 解答状況が悪い学生を抽出した. このような学生に対してメールを送るなどの対応を行う予定であったが，授業担当者の多忙により 1 回しか対応ができなかった．このため，解答状況が少ない学生へのメッセージの自動送信機能が望まれる.

2017 年度には解答期限を設けたが, 期限内に解答したかどうかは，定期的に解答履歴を取得して確認した．解答期日によるログのフィルタリング機能を追加することで学生の期限内の解答状況を確認できると考える．また，問題の解答期限を設定し，解答が遅れた問題に遅延マーク，新規に出題された問題に New マークを付加することで, 学生自身が進捗状況を視覚的に把握できると考える.

また，pgtracer は解答を提出すると正誤と正答が表示される．しかし，正答が表示されるだけでは，学生がなぜ間違ったのか, 正答を導く思考の過程に気づくことが難しい. そこで，正答とあわせて解説も表示することで，学生の理解を促進できると考える.

トレース表の表示において，トレース表の横幅が大きい場合に，一部が表示されず，ブラウザの縮小機能を用いて全体を表示している．現在は，変数名や関数名の名前を短くする，配列を小さくするなどして対応しているが，UI による工夫についても検討したい.

問題作成の点からは，作成した XML ファイルに対する削除，名前の変更，サブフォルダ等を用いたファイルの整理機能が望まれる．また，マスクを設定する際に，マスク部分を指定した後，マスク設定ボタンを押下するが，マスク部分とボタンの位置を近づけることで問題作成の効率が向上すると考える.

7. おわりに

本稿では，pgtracer を自学習に活用したプログラミング科目の実践について報告した．自学習の成果を科目の成績に反映することによって，学生の学習行動が向上すること

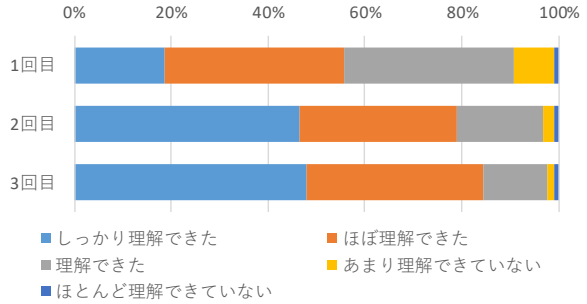


図 6　トレース表の理解

Figure 6　Understanding of Trace Table

が分かった．また，授業中の説明や演習時にトレース表を用いることで，トレース表の理解が向上する.

今後の課題としては，6.3 節で述べた pgtracer の機能拡張について検討するとともに，収集した学習履歴を分析し，学生の理解度を客観的に推測するための手法について検討したい．また，2018 年度前期開講科目「プログラミング基礎 II」においても自習課題の提供を実践している．そこでの解答を確認したところ，プログラムの穴埋めにおいて，本来ならば変数を解答すべきところで，定数を解答して正解となっている解答を見つけた．このような解答をする学生は，プログラムを理解しているとは言えない．今後このような解答した学生の解答状況について精査し，プログラミング能力との関連を検討していきたい．また，このような解答を誤りとするために，複数の入力値を用いて実行結果が一致することを確認するなどの手法を検討したい.

謝辞　本研究は JSPS 科研費 17K01036 の助成を受けたものです.

参考文献

[1] 林文部科学大臣，Society5.0 に向けた人材育成の推進，未来投資会議資料，2018 年 5 月. http://www.kantei.go.jp/jp/singi/keizaisaisei/miraitoshikaigi/dai16/siryou6.pdf

[2] 掛下哲郎，柳田峻，太田康介，“穴埋め問題を用いたプログラミング教育支援ツール pgtracer の開発と評価”，情報処理学会論文誌：教育とコンピュータ, Vol. 2, No. 2, pp. 20-36，2016 年 10 月.

[3] Miyuki Murata, Tetsuro Kakeshita, "Analysis Method of Student Achievement Level utilizing Web-Based Programming Education Support Tool pgtacer", 5th International Conference on Learning Technologies and Learning Environment (LTLE 2016), pp.316-321, July 2016.

[4] T. Kakeshita, M. Murata, “Application of Programming Education Support Tool pgtracer for Homework Assignment”, International Journal of Learning Technologies and Learning Environments, Val. 1, No. 1, pp. 40-61, 2018.